CHAPTER 6
Hypothesis and Significance Testing

Introduction

Testing a claim and drawing conclusion from the result is testing association. It is one of the most done work in statistics. For which hypothesis testing defines a claim and using significance level and bunch of different tests. The validity of the claim in relation to the data is checked. Hypothesis testing is a method of making decisions based on data analysis. It involves stating a null hypothesis and an alternative hypothesis, which are mutually exclusive statements about a population parameter. Significance tests are procedures that assess how likely it is that the observed data are consistent with the null hypothesis. There are different types of statistical tests that can be used for hypothesis testing, depending on the nature of the data and the research question. Such as z-test, t-test, chi-square test, ANOVA. These are described later in the chapter, with examples. Sampling techniques and sampling distributions are important concepts, and sometimes they are critical in hypothesis testing because they affect the validity and reliability of the results. Sampling techniques are methods

of selecting a subset of individuals or units from a population that is intended to be representative of the population. Sampling distributions are the probability distributions of the possible values of a sample statistic based on repeated sampling from the population.

Structure

In this chapter, we will discuss the following topics:

- Hypothesis testing
- Significance tests
- Role of p-value and significance level
- Statistical test
- Sampling techniques and sampling distributions

Objectives

The objective of this chapter is to introduce the concept of hypothesis testing, determining significance, and interpreting hypotheses through multiple testing. A hypothesis is a claim or technique for drawing a conclusion, and a significance test checks the likelihood that the claim or conclusion is correct. We will see how to perform them and interpret the result obtained from the data. This chapter also discusses the types of tests used for hypothesis testing and significance testing. In addition, this chapter will explain the role of the p-value and the significance level. Finally, this chapter shows how to use various hypothesis and significance tests and p-values to test hypotheses.

Hypothesis testing

Hypothesis testing is a statistical method that uses data from a sample to draw conclusions about a population. It involves testing an assumption, known as the null hypothesis, to determine whether it is likely to be true or

false. The null hypothesis typically states that there is no effect or difference between two groups, while the alternative hypothesis is the opposite and what we aim to prove. Hypothesis testing checks if an idea about the world is true or not. For example, you might have an idea that men are taller than women on average, and you want to see if the data support your idea or not.

Tutorial 6.1: An illustration of the hypothesis testing using the example 'men are taller than women on average', as mentioned in above example, is as follows:

```
import scipy.stats as stats
# define the significance level
# alpha = 0.05, which means there is a 5% chance of making a type I error (rejecting the null hypothesis when it is true)
alpha = 0.05
# generate some random data for men and women heights (in cm)
# you can replace this with your own data
men_heights = stats.norm.rvs(loc=175, scale=10, size=100) # mean = 175, std = 10
women_heights = stats.norm.rvs(loc=165, scale=8, size=100) # mean = 165, std = 8
# calculate the sample means and standard deviations
men_mean = men_heights.mean()
men_std = men_heights.std()
women_mean = women_heights.mean()
women_std = women_heights.std()
# print the sample statistics
print("Men: mean = {:.2f}, std = {:.2f}".format(men_mean, men_std))
print("Women: mean = {:.2f}, std = {:.2f}".format(women_mean, women_std))
```

```
# perform a two-sample t-test
# the null hypothesis is that the population means are e
qual
# the alternative hypothesis is that the population means
 are not equal
t_stat, p_value = stats.ttest_ind(men_heights, women_hei
ghts)
# print the test statistic and the p-value
print("t-statistic = {:.2f}".format(t_stat))
print("p-value = {:.4f}".format(p_value))
# compare the p-
value with the significance level and make a decision
if p_value <= alpha:
    print("Reject the null hypothesis: the population mean
s are not equal.")
else:
    print("Fail to reject the null hypothesis: the populatio
n means are equal.")
```

Output: Number and result may vary based on a random generated number. Following is the snippet of output:

```
Men: mean = 174.48, std = 9.66
Women: mean = 165.16, std = 7.18
t-statistic = 7.70
p-value = 0.0000
Reject the null hypothesis: the population means are not
equal.
```

Here is a simple explanation of how hypothesis testing works. Suppose you have a jar of candies, and you want to determine whether there are more red candies than blue candies in the jar. Since counting all the candies in the jar is not feasible, you can extract a handful of them and determine the number of red and blue candies. This process is known as **sampling**. Based on the sample, you can make

an inference about the entire jar. This inference is referred to as a **hypothesis**, which is akin to a tentative answer to a question. However, to determine the validity of this hypothesis, a comparison between the sample and the expected outcome is necessary. For instance, consider the hypothesis: *There are more red candies than blue candies in the jar.* This comparison is known as a **hypothesis test**, which determines the likelihood of the sample matching the hypothesis. For instance, if the hypothesis is correct, the sample should contain more red candies than blue candies. However, if the hypothesis is incorrect, the sample should contain roughly the same number of red and blue candies. A test provides a numerical measurement of how well the sample aligns with the hypothesis. This measurement is known as a **p-value**, which indicates the level of surprise in the sample. A low p-value indicates a highly significant result, while a high p-value indicates a result that is not statistically significant. For instance, if you randomly select a handful of candies and they are all red, the result would be highly significant, and the p-value would be low. However, if you randomly select a handful of candies and they are half red and half blue, the result would not be statistically significant, and the p-value would be high. Based on the p-value, one can determine whether the hypothesis is true or false. This determination is akin to a final answer to the question. For instance, if the p-value is low, it can be concluded that the hypothesis is true, and one can state that there are more red candies than blue candies in the jar. Conversely, if the p-value is high, it can be concluded that the hypothesis is false, and one can state: The jar does not contain more red candies than blue candies.

Tutorial 6.2: An illustration of the hypothesis testing using the example *jar of candies,* as mentioned in above example, is as follows:

```
# import the scipy.stats library
import scipy.stats as stats
# define the significance level
alpha = 0.05
# geerate some random data for the number of red and blue candies in a handful
# you can replace this with your own data
n = 20 # number of trials (candies)
p = 0.5 # probability of success (red candy)
red_candies = stats.binom.rvs(n, p) # number of red candies
blue_candies = n - red_candies # number of blue candies
# print the sample data
print("Red candies: {}".format(red_candies))
print("Blue candies: {}".format(blue_candies))
# perform a binomial test
# the null hypothesis is that the probability of success is 0.5
# the alternative hypothesis is that the probability of success is not 0.5
p_value = stats.binomtest(red_candies, n, p, alternative='two-sided')
# print the p-value
print("p-value = {:.4f}".format(p_value.pvalue))
# compare the p-value with the significance level and make a decision
if p_value.pvalue <= alpha:
    print("Reject the null hypothesis: the probability of success is not 0.5.")
else:
    print("Fail to reject the null hypothesis: the probability of success is 0.5.")
```

Output: Number and result may vary based on generated

random number. Following is the snippet of output:

```
Red candies: 6
Blue candies: 14
p-value = 0.1153
Fail to reject the null hypothesis: the probability of success is 0.5.
```

Steps of hypothesis testing

Following are the steps to perform hypothesis testing:

1. State your null and alternate hypothesis. Keep in mind that the null hypothesis is what you assume to be true before you collect any data, while the alternate hypothesis is what you want to prove or test. For instance, if you aim to test whether men are taller than women on average, your null hypothesis could be: **There is no significant difference in height between men and women**. The alternate hypothesis could be: **On average, men are taller than women**.

In *Tutorial 6.1*, the following snippet states hypothesis:

```
# the null hypothesis is that the population means are equal
# the alternative hypothesis is that the population means are not equal
t_stat, p_value = stats.ttest_ind(men_heights, women_heights)
```

In *Tutorial 6.2,* the following snippet states hypothesis:

```
# the null hypothesis is that the probability of success is 0.5
# the alternative hypothesis is that the probability of success is not 0.5
p_value = stats.binomtest(red_candies, n, p, alternative='two-sided')
```

2. Collect data in a way that is designed to test your

hypothesis. For example, you might measure the heights of a random sample of men and women from different regions and social classes.

In *Tutorial 6.1,* the following snippet generates 100 random samples of heights from a normal distribution with a specified mean (**loc**) and a standard deviation (**scale**)*:*

```
men_heights = stats.norm.rvs(loc=175, scale=10, size=100) # mean = 175, std = 10
women_heights = stats.norm.rvs(loc=165, scale=8, size=100) # mean = 165, std = 8
```

In *Tutorial 6.2*, the following snippet generates random number of candies based on scenario where there are 20 candies, each with a 50% chance of being red:

```
n = 20 # number of trials (candies)
p = 0.5 # probability of success (red candy)
red_candies = stats.binom.rvs(n, p) # number of red candies
blue_candies = n - red_candies # number of blue candies
```

3. Perform a statistical test that compares your data with your null hypothesis. It's crucial to choose the appropriate statistical test based on the nature of your data and the objective of your study, which are described in the *Statistical test* section below. For example, you might use a t-test to see if the average height of men is different from the average height of women in your sample.

In *Tutorial 6.1,* the following snippet performs a test to compute the t-statistic and p-value:

```
t_stat, p_value = stats.ttest_ind(men_heights, women_heights)
```

In *Tutorial 6.2***,** the following snippet performs a binomial test to compute the p-value:

```
p_value = stats.binom_test(red_candies, n, p, alternative='two-sided')
```

4. Decide whether to reject or fail to reject your null hypothesis based on your test result. For instance, you can use a significance level of 0.05. This means you are willing to accept a 5% chance of being wrong. If your p-value is less than 0.05, you can reject your null hypothesis and accept your alternate hypothesis. If your p-value is more than 0.05, you cannot reject your null hypothesis and must keep it.

In *Tutorial 6.1***.** the following snippet checks the hypothesis based on the p-value:

```
if p_value <= alpha:
    print("Reject the null hypothesis: the population means are not equal.")
else:
    print("Fail to reject the null hypothesis: the population means are equal.")
```

In *Tutorial 6.2,* the following snippet checks the hypothesis based on the p-value.

```
if p_value.pvalue <= alpha:
    print("Reject the null hypothesis: the probability of success is not 0.5.")
else:
    print("Fail to reject the null hypothesis: the probability of success is 0.5.")
```

5. Present your findings. For instance, you can report the mean and standard deviation of the heights of men and women in your sample, the t-value and p-value of your test, and your conclusion regarding the hypothesis. In *Tutorial 6.1* and *Tutorial 6.2* all the **print** statement present the findings.

Types of hypothesis testing

There are various types of hypothesis testing, depending on the number and nature of the hypotheses and the data.

Some common types include:

- **One-sided and two-sided tests**: A one-tailed test is when you have a specific direction for your alternative hypothesis, such as *men are on average taller than women*. A two-tailed test is when you have a general direction for your alternative hypothesis, such as *men and women have different average heights*.

  For example, suppose you want to know if your class (Class 1) is smarter than another class (Class 2). You could give both classes a math test and compare their scores. A one-tailed test is when you are only interested in one direction, such as my class (Class 1) is smarter than the other class (Class 2). A two-tailed test is when you are interested in both directions, such as Class 1 and the Class 2 are different in smartness.

Tutorial 6.3: An illustration of the one-sided testing using the example *my class (Class 1) is smarter than the other class (Class 2)*, as mentioned in above example, is as follows:

```
# Import the scipy.stats module
import scipy.stats as stats
# Define the scores of both classes as lists
class1 = [80, 85, 90, 95, 100, 105, 110, 115, 120, 125]
class2 = [75, 80, 85, 90, 95, 100, 105, 110, 115, 120]
# Perform a one-sided test to see if class1 is smarter than class2
# The null hypothesis is that the mean of class1 is less than or
equal to the mean of class2
# The alternative hypothesis is that the mean of class1 is greater than the mean of class2
t_stat, p_value = stats.ttest_ind(class1, class2, alternative='greater')
print('One-sided test results:')
```

```
print('t-statistic:', t_stat)
print('p-value:', p_value)
# Compare the p-value with the significance level
if p_value < 0.05:
    print('We reject the null hypothesis and conclude that class1 is smarter than class2.')
else:
    print('We fail to reject the null hypothesis and cannot conclude that class1 is smarter than class2.')
```

Output:

```
One-sided test results:
t-statistic: 0.7385489458759964
p-value: 0.2348510364004045
We fail to reject the null hypothesis and cannot conclude that class1 is smarter than class2.
```

Tutorial 6.4: An illustration of the two-sided testing using the example *my class (Class 1) and the other class (Class 2) are different in smartness*, as mentioned in above example, is as follows:

```
# Import the scipy.stats module
import scipy.stats as stats
# Define the scores of both classes as lists
class1 = [80, 85, 90, 95, 100, 105, 110, 115, 120, 125]
class2 = [75, 80, 85, 90, 95, 100, 105, 110, 115, 120]
# Perform a two-sided test to see if class1 and class2 are different in smartness
# The null hypothesis is that the mean of class1 is equal to the mean of class2
# The alternative hypothesis is that the mean of class1 is not equal to the mean of class2
t_stat, p_value = stats.ttest_ind(class1, class2, alternative='two-sided')
```

```
print('Two-sided test results:')
print('t-statistic:', t_stat)
print('p-value:', p_value)
# Compare the p-value with the significance level
if p_value < 0.05:
    print('We reject the null hypothesis and conclude that class1 and class2 are different in smartness.')
else:
    print('We fail to reject the null hypothesis and cannot conclude that class1 and class2 are different in smartness.')
```

Output:

```
Two-sided test results:
t-statistic: 0.7385489458759964
p-value: 0.4697020728008009
We fail to reject the null hypothesis and cannot conclude that class1 and
class2 are different in smartness.
```

- **One-sample and two-sample tests**: A one-sample test is when you compare a single sample to a known population value, such as *the average height of men in Norway is 180 cm*. A two-sample test is when you compare two samples, such as *the average height of men in Norway is different from the average height of men in Japan*.

For example, imagine you want to know if a class is taller than the average height for kids of their age. You can measure the heights of everyone in the class and compare them to the average height of kids of their age. A one-sample test is when you have only one group of data, such as *my class is taller than the average height for kids my age*. A two-sample test is when you have two groups of data, such as *my class is taller than the other*

class.

Tutorial 6.5: An illustration of the one-sample testing using the example *my class (Class 1) is taller than the average height for kids my age*, as mentioned in above example, is as follows:

```
# Import the scipy.stats module
import scipy.stats as stats
# Define the heights of your class as a list
my_class = [150, 155, 160, 165, 170, 175, 180, 185, 190, 195]
# Perform a one-sample test to see if your class is taller than the average height for kids your age
# The null hypothesis is that the mean of your class is equal to the population mean
# The alternative hypothesis is that the mean of your class is not equal to the population mean (two-sided)
# or that the mean of your class is greater than the population mean (one-sided)
# According to the WHO, the average height for kids aged 12 years is 152.4 cm for boys and 151.3 cm for girls [^1^][1]
# We will use the average of these two values as the population mean
pop_mean = (152.4 + 151.3) / 2
t_stat, p_value = stats.ttest_1samp(my_class, pop_mean, alternative='two-sided')
print('One-sample test results:')
print('t-statistic:', t_stat)
print('p-value:', p_value)
# Compare the p-value with the significance level
if p_value < 0.05:
    print('We reject the null hypothesis and conclude that
```

```
    your class is different in height from the average height for kids your age.')
else:
    print('We fail to reject the null hypothesis and cannot conclude that your class is different in height from the a verage height for kids your age.')
```

Output:

```
One-sample test results:
t-statistic: 4.313644314582188
p-value: 0.0019512458685808432
We reject the null hypothesis and conclude that your cla ss is different in height from the average height for kids your age.
```

Tutorial 6.6: An illustration of the two-sample testing using the example *my class (Class 1) is taller than the other class (Class 2)*, as mentioned in above example, is as follows:

```
# Import the scipy.stats module
import scipy.stats as stats
# Define the heights of your class as a list
my_class = [150, 155, 160, 165, 170, 175, 180, 185, 190, 195]
# Perform a two-
sample test to see if your class is taller than the other cl ass
# The null hypothesis is that the means of both classes a re equal
# The alternative hypothesis is that the means of both cl asses are not equal (two-sided)
# or that the mean of your class is greater than the mea n of the other class (one-sided)
# Define the heights of the other class as a list
other_class = [145, 150, 155, 160, 165, 170, 175, 180, 1 85, 190]
```

```
t_stat, p_value = stats.ttest_ind(my_class, other_class, alternative='two-sided')
print('Two-sample test results:')
print('t-statistic:', t_stat)
print('p-value:', p_value)
# Compare the p-value with the significance level
if p_value < 0.05:
    print('We reject the null hypothesis and conclude that your class and the other class are different in height.')
else:
    print('We fail to reject the null hypothesis and cannot conclude that your class and the other class are different in height.')
```

Output:

```
Two-sample test results:
t-statistic: 0.7385489458759964
p-value: 0.4697020728008009
We fail to reject the null hypothesis and cannot conclude that your class and the other class are different in height.
```

- **Paired and independent tests**: A paired test is when you compare two samples that are related or matched in some way, such as *the average height of men before and after growth hormone treatment*. An independent test is when you compare two samples that are unrelated or random, such as *the average height of men and women*.

For example, imagine you want to know if your class is happier after a field trip. You could ask everyone in your class to rate their happiness before and after the field trip and compare their ratings. A paired test is when you have two sets of data that are linked or matched, such as *my happiness before and after the field trip*. An independent test is when you have two sets of data that

are not linked or matched, such as *my happiness and the happiness of the other class*.

Tutorial 6.7: An illustration of the paired testing using the example *my happiness before and after the field trip,* as mentioned in above example, is as follows:

```
# We use scipy.stats.ttest_rel to perform a paired t-test
# We assume that the happiness ratings are on a scale of 1 to 10
import scipy.stats as stats
# The happiness ratings of the class before and after the field trip
before = [7, 8, 6, 9, 5, 7, 8, 6, 7, 9]
after = [8, 9, 7, 10, 6, 8, 9, 7, 8, 10]
# Perform the paired t-test
t_stat, p_value = stats.ttest_rel(before, after)
# Print the results
print("Paired t-test results:")
print("t-statistic:", t_stat)
print("p-value:", p_value)
```

Output:

```
Paired t-test results:
t-statistic: -inf
p-value: 0.0
```

Tutorial 6.8: An illustration of the independent test using the example *my happiness and the happiness of the other class,* as mentioned in above example, is as follows:

```
# We use scipy.stats.ttest_ind to perform an independent t-test
# We assume that the happiness ratings of the other class are also on a scale of 1 to 10
import scipy.stats as stats
# The happiness ratings of the other class before and aft
```

```
er the field trip
other_before = [6, 7, 5, 8, 4, 6, 7, 5, 6, 8]
other_after = [7, 8, 6, 9, 5, 7, 8, 6, 7, 9]
# Perform the independent t-test
t_stat, p_value = stats.ttest_ind(after, other_after)
# Print the results
print("Independent t-test results:")
print("t-statistic:", t_stat)
print("p-value:", p_value)
```

Output:

```
Independent t-test results:
t-statistic: 1.698415551216892
p-value: 0.1066484282683789 2
```

- **Parametric and nonparametric tests**: A parametric test is when you assume that your data follow a certain distribution, such as a normal distribution, and you use parameters such as mean and standard deviation to describe your data. A nonparametric test is when you do not assume that your data follow a particular distribution, and you use ranks or counts to describe your data.

  For example, imagine you want to know if your class likes chocolate or vanilla ice cream more. You could ask everyone in your class to choose their favorite flavor and count how many people like each flavor. A parametric test is when you assume that your data follow a pattern or shape, such as a bell curve, and you use numbers like mean and standard deviation to describe your data. A nonparametric test is when you do not assume that your data follow a pattern or shape, and you use ranks or counts to describe your data.

Tutorial 6.9: An illustration of the parametric test, as mentioned in above example, is as follows:

```
# We use scipy.stats.ttest_ind to perform a parametric t-
```

```
test
# We assume that the data follows a normal distribution
import scipy.stats as stats
# The number of students who like chocolate and vanilla ice cream
chocolate = [25, 27, 29, 28, 26, 30, 31, 24, 27, 29]
vanilla = [22, 23, 21, 24, 25, 26, 20, 19, 23, 22]
# Perform the parametric t-test
t_stat, p_value = stats.ttest_ind(chocolate, vanilla)
# Print the results
print("Parametric t-test results:")
print("t-statistic:", t_stat)
print("p-value:", p_value)
```

Output:

```
Parametric t-test results:
t-statistic: 5.190169516378603
p-value: 6.162927154861931e-05
```

Tutorial 6.10: An illustration of the nonparametric test, as mentioned in above example, is as follows:

```
# We use scipy.stats.mannwhitneyu to perform a nonparametric Mann-Whitney U test
# We do not assume any distribution for the data
import scipy.stats as stats
# The number of students who like chocolate and vanilla ice cream
chocolate = [25, 27, 29, 28, 26, 30, 31, 24, 27, 29]
vanilla = [22, 23, 21, 24, 25, 26, 20, 19, 23, 22]
# Perform the nonparametric Mann-Whitney U test
u_stat, p_value = stats.mannwhitneyu(chocolate, vanilla)
# Print the results
print("Nonparametric Mann-Whitney U test results:")
print("U-statistic:", u_stat)
```

```
print("p-value:", p_value)
```

Output:

```
Nonparametric Mann-Whitney U test results:
U-statistic: 95.5
p-value: 0.0006480405677249192
```

Significance testing

Significance testing evaluates the likelihood of a claim or statement about a population being true using data. For instance, it can be used to test if a new medicine is more effective than a placebo or if a coin is biased. The p-value is a measure used in significance testing that indicates how frequently you would obtain the observed data or more extreme data if the claim or statement were false. The smaller the p-value, the stronger the evidence against the claim or statement. Significance testing is different from hypothesis testing, although they are often confused and used interchangeably. Hypothesis testing is a formal procedure for comparing two competing statements or hypotheses about a population, and making a decision based on the data. One of the hypotheses is called the **null hypothesis**, the other hypothesis is called the **alternative hypothesis**, as described above in **hypothesis testing**. Hypothesis testing involves choosing a significance level, which is the maximum probability of making a wrong decision when the null hypothesis is true. Usually, the significance level is set to 0.05. Hypothesis testing also involves calculating a test statistic, which is a number that summarizes the data and measures how far it is from the null hypothesis. Based on the test statistic, a p-value is computed, which is the probability of getting the data (or more extreme) if the null hypothesis is true. If the p-value is less than the significance level, the null hypothesis is rejected and the alternative hypothesis is accepted. If the p-

value is greater than the significance level, the null hypothesis is not rejected and the alternative hypothesis is not accepted.

Suppose, you have a friend who claims to be able to guess the outcome of a coin toss correctly more than half the time, you can test their claim using significance testing. Ask them to guess the outcome of 10-coin tosses and record how many times they are correct. If the coin is fair and your friend is just guessing, you would expect them to be right about 5 times out of 10, on average. However, if they get 6, 7, 8, 9, or 10 correct guesses, how likely is it to happen by chance? The p-value answers the question of the probability of getting the same or more correct guesses as your friend did, assuming a fair coin and random guessing. A smaller p-value indicates a lower likelihood of this happening by chance, and therefore raises suspicion about your friend's claim. Typically, a p-value cutoff of 0.05 is used. If the p-value is less than 0.05, we consider the result statistically significant and reject the claim that the coin is fair, and the friend is guessing. If the p-value is greater than 0.05, we consider the result not statistically significant and do not reject the claim that the coin is fair, and the friend is guessing.

Tutorial 6.11: An illustration of the significance testing, based on above coin toss example, is as follows:

```
1. # Import the binom_test function from scipy.stats
2. from scipy.stats import binomtest
3. # Ask the user to input the number of correct guesses b
   y their friend
4. correct = int(input("How many correct guesses did your
   friend make out of 10 coin tosses? "))
5. # Calculate the p-value using the binom_test function
6. # The arguments are: number of successes, number of tr
   ials,
   probability of success, alternative hypothesis
```

```
p_value = binomtest(correct, 10, 0.5, "greater")
# Print the p-value
print("p-value = {:.4f}".format(p_value.pvalue))
# Compare the p-value with the cutoff of 0.05
if p_value.pvalue < 0.05:
    # If the p-value is less than 0.05, reject the claim that the coin is fair and the friend is guessing
    print("This result is statistically significant. We reject the claim that the coin is fair and the friend is guessing.")
else:
    # If the p-value is greater than 0.05, do not reject the claim that the coin is fair and the friend is guessing
    print("This result is not statistically significant. We do not reject the claim that the coin is fair and the friend is guessing.")
```

Output: For nine correct guesses, is as follows:

```
How many correct guesses did your friend make out of 10 coin tosses? 9
p-value = 0.0107
This result is statistically significant.
We reject the claim that the coin is fair and the friend is guessing.
```

For two correct guesses, the output is not statistically significant as follows:

```
How many correct guesses did your friend make out of 10 coin tosses? 2
p-value = 0.9893
This result is not statistically significant. We do not reject the claim that the coin
is fair and the friend is guessing.
```

The following is another example to better understand the relation between hypothesis and significance testing. Suppose, you want to know whether a new candy makes children smarter. You have two hypotheses: The null hypothesis is that *the candy has no effect on children's intelligence*. The alternative hypothesis is that *the candy increases children's intelligence*.

You decide to test your hypotheses by giving the candy to 20 children and a placebo to another 20 children. You then measure their IQ scores before and after the treatment. You choose a significance level of 0.05, meaning that you are willing to accept a 5% chance of being wrong if the candy has no effect. You calculate a test statistic, which is a number that tells you how much the candy group improved compared to the placebo group. Based on the test statistic, you calculate a p-value, which is the probability of getting the same or greater improvement than you observed if the candy had no effect.

If the p-value is less than 0.05, you reject the null hypothesis and accept the alternative hypothesis. You conclude that the candy makes the children smarter.

If the p-value is greater than 0.05, you do not reject the null hypothesis and you do not accept the alternative hypothesis. You conclude that the candy has no effect on the children's intelligence.

Tutorial 6.12: An illustration of the significance testing, based on above candy and smartness example, is as follows:

```
# Import the ttest_rel function from scipy.stats
from scipy.stats import ttest_rel
# Define the IQ scores of the candy group before and aft
er the treatment
candy_before = [100, 105, 110, 115, 120, 125, 130, 135,
140]
candy_after = [104, 105, 110, 120, 123, 125, 135, 135, 1
```

```
44]
# Define the IQ scores of the placebo group before and after the treatment
placebo_before = [101, 106, 111, 116, 121, 126, 131, 136, 141]
placebo_after = [100, 104, 109, 113, 117, 121, 125, 129, 133]
# Calculate the difference in IQ scores for each group
candy_diff = [candy_after[i] - candy_before[i] for i in range(9)]
placebo_diff = [placebo_after[i] - placebo_before[i] for i in range(9)]
# Perform a paired t-test on the difference scores
# The null hypothesis is that the mean difference is zero
# The alternative hypothesis is that the mean difference is positive
t_stat, p_value = ttest_rel(candy_diff, placebo_diff, alternative="greater")
# Print the test statistic and the p-value
print(f"The test statistic is {t_stat:.4f}")
print(f"The p-value is {p_value:.4f}")
# Compare the p-value with the significance level of 0.05
if p_value < 0.05:
    # If the p-value is less than 0.05, reject the null hypothesis and accept the alternative hypothesis
    print("This result is statistically significant. We reject the null hypothesis and accept the alternative hypothesis.")
    print("We conclude that the candy makes the children smarter.")
else:
```

```
    # If the p-
value is greater than 0.05, do not reject the null hypothe
sis and do not accept the alternative hypothesis
    print("This result is not statistically significant. We do
not reject the null hypothesis and do not accept the alter
native hypothesis.")
    print("We conclude that the candy has no effect on th
e
children's intelligence.")
```

Output:

```
The test statistic is 5.6127
The p-value is 0.0003
This result is statistically significant.
We reject the null hypothesis and accept the alternative
hypothesis.
We conclude that the candy makes the children smarter.
```

The above output can be changed by changing the p-value, as indicated. The p-value depends on the before and after values.

Steps of significance testing

The steps to perform significance testing in statistics is described by the example below:

- **Question**: Does drinking coffee make you more alert than drinking water?
- **Guess**: There is no difference in alertness between coffee and water. Coffee will make you more alert than water.
- **Chance**: 5%, meaning you are willing to accept a 5% chance of being wrong if there is no difference in alertness between coffee and water.
- **Number**: Suppose -3.2 is test statistic, based on the difference in average alertness scores between two groups of 20 students each who drank coffee or water

before taking a test. The assumed mean scores are 75 and 80, and the standard deviations are 10 and 12, respectively.

- **Probability**: 0.003, which is the probability of getting the same or greater difference in scores than you observed if there is no difference in alertness between coffee and water.
- **Decision**: Since the probability is less than chance, you do not believe the conjecture that there is no difference in alertness between coffee and water, and you believe the conjecture that coffee makes you more alert than water.
- **Answer**: You have strong evidence that coffee makes you more alert than water, with a 5% chance of being wrong. The average difference in alertness is -5, with a assumed range of (-8.6, -1.4).

Further explanation of significance testing along with the *candy makes the children smarter* example, is as follows:

1. **State the claim or statement that you want to test**: This is usually the research question or the effect of interest.

   Claim: A new candy makes the children smarter.

   State the null and alternative hypotheses. The null hypothesis is the opposite of the claim or statement, and it usually represents no effect or no difference. The alternative hypothesis is the same as the claim or statement, and it usually represents the effect or difference of interest as follows:

   - **Null hypothesis**: The candy has no effect on the children’s intelligence, so the mean difference is zero.
   - **Alternative hypothesis**: The candy increases the children’s intelligence, so the mean difference is positive.

In *Tutorial 6.12*, the following snippet states the claim and hypothesis:

```
# The null hypothesis is that the mean difference is zero
# The alternative hypothesis is that the mean difference is positive
t_stat, p_value = ttest_rel(candy_diff, placebo_diff, alternative="greater")
```

2. **Choose a significance level**: This is the maximum probability of rejecting the null hypothesis when it is true. Usually, the significance level is set to 0.05, but it can be higher or lower depending on the context and the consequences of making a wrong decision.

   Significance level: 0.05

3. **Choose and compute a test statistic and p-value**: This is a number that summarizes the data and measures how far it is from the null hypothesis. Different types of data and hypotheses require different types of test statistics, such as z, t, F, or chi-square. The test statistic depends on the sample size, the sample mean, the sample standard deviation, and the population parameters.

   Test statistic: test statistic is 5.6127.

   P-value is the probability of getting the data (or more extreme) if the null hypothesis is true. The p-value depends on the test statistic and the distribution that it follows under the null hypothesis. The p-value can be calculated using formulas, tables, or software.

   P-value: p-value is 0.0003.

   In *Tutorial 6.12*, the following snippet computes the p-value and test statistic:

```
t_stat, p_value = ttest_rel(candy_diff, placebo_diff, alternative="greater")
# Print the test statistic and the p-value
```

```
print(f"The test statistic is {t_stat:.4f}")
print(f"The p-value is {p_value:.4f}")
```

4. **Compare the p-value to the significance level and decide**: If the p-value is less than the significance level, reject the null hypothesis and accept the alternative hypothesis. If the p-value is greater than the significance level, do not reject the null hypothesis and do not accept the alternative hypothesis.

   Decision: Since the p-value is less than the significance level, reject the null hypothesis and accept the alternative hypothesis.

   In *Tutorial 6.12***,** the following snippet compares p-value and significance level:

```
# Compare the p-
value with the significance level of 0.05
if p_value < 0.05:
    # If the p-value is less than 0.05, reject the null
hypothesis and accept the alternative hypothesis
    print("This result is statistically significant. We
reject the null hypothesis and accept the alternative
hypothesis.")
    print("We conclude that the candy makes the
children smarter.")
else:
    # If the p-value is greater than 0.05, do not
reject the null hypothesis and do not accept the
alternative hypothesis
    print("This result is not statistically
significant. We do not reject the null hypothesis and
do not
accept the alternative hypothesis.")
    print("We conclude that the candy has no
effect on the children's intelligence.")
```

5. **Interpret the results and draw conclusions**: Explain what the decision means in the context of the problem and the data. Address the original claim or statement and the effect of interest. Report the test statistic, the p-value, and the significance level. Discuss the limitations and assumptions of the analysis and suggest possible directions for further research.

Summary: There is sufficient evidence to conclude that the new candy makes children smarter, at the 0.05 significance level.

Types of significance testing

Depending on the data and the hypotheses you want to test, there are different types. Some common types are as follows:

- **T-test**: Compares the means of two independent samples with a continuous dependent variable. For example, you might use a t-test to see if there is a difference in blood pressure (continuous dependent variable) between patients taking a new drug and those taking a placebo.
- **ANOVA**: Compare the means of more than two independent samples with a continuous dependent variable. For example, you can use ANOVA to see if there is a difference in test scores (continuous dependent variable) between students who study using different methods.
- **Chi-square test**: Evaluate the relationship between two categorical variables. For example, you can use a chi-squared test to see if there is a relationship between gender (male/female) and voting preference (A party/B party).
- **Correlation test**: Measures the strength and direction of a linear relationship between two continuous variables. For example, you can use a correlation test to see how height and weight are related.

- **Regression test**: Estimate the effect of one or more predictor (independent) variables on an outcome (dependent) variable. For example, you might use a regression test to see how age, education, and income affect life satisfaction.

Role of p-value and significance level

P-values and **significance levels** are tools that helps to decide whether to reject the null hypothesis. A p-value is the probability of getting the data you observe, or more extreme data, if the null hypothesis is true. A significance level is a threshold you choose before the test, usually 0.05 or 0.01.

To illustrate these concepts, consider the example of coin flipping. Suppose, you want to test whether a coin is fair, meaning that it has a 50% chance of landing heads or tails. The null hypothesis is that the coin is fair, and the alternative hypothesis is that the coin is not fair. You decide to flip the coin 10 times and count the number of heads. You also choose a significance level of 0.05 for the test. A significance level of 0.05 indicates that there is a 5% risk of rejecting the null hypothesis if it is true. In other words, you are willing to accept a 5% chance of reaching the wrong conclusion.

You flip the coin 10 times and get 8 heads and 2 tails. Is this result unusual if the coin is fair? To answer this question, you need to calculate the p-value. The p-value is the probability of getting 8 or more heads in 10 flips if the coin is fair. You can use a binomial calculator to find this probability. The p-value is 0.0547, which means that there is a 5.47% chance of getting 8 or more heads in 10 flips when the coin is fair. Now, compare the p-value with the significance level. The p-value is 0.0547, which is slightly greater than the significance level of 0.05. This means that you cannot reject the null hypothesis. You have to say that the data is not enough to prove that the coin is not fair.

Maybe you just got lucky with the tosses, or maybe you need more data to detect a difference.

Tutorial 6.13: To compute the p-value of getting 8 heads and 2 tails when a coin is flipped 10 times, with a significance level of 0.05, as in the example above, is as follows:

```
# Import the scipy library for statistical functions
import scipy.stats as stats
# Define the parameters of the binomial distribution
n = 10 # number of flips
k = 8 # number of heads
p = 0.5 # probability of heads
# Calculate the p-
value using the cumulative distribution function (cdf)
# The p-
value is the probability of getting at least k heads, so we
use 1 - cdf(k-1)
p_value = 1 - stats.binom.cdf(k-1, n, p)
# Print the p-value
print(f"The p-value is {p_value:.4f}")
# Compare the p-value with the significance level
alpha = 0.05 # significance level
if p_value < alpha:
    print("The result is statistically significant.")
else:
    print("The result is not statistically significant.")
```

Output:

```
The p-value is 0.0547
The result is not statistically significant.
```

The result means that the outcome of the experiment (8 heads and 2 tails) is not very unlikely to occur by chance, assuming the coin is fair. In other words, there is not

enough evidence to reject the null hypothesis that the coin is fair.

Statistical tests

Commonly used statistical tests include the z-test, t-test, and chi-square test, which are typically applied to different types of data and research questions. Each of these tests plays a crucial role in the field of statistics, providing a framework for making inferences and drawing conclusions from data. Z-test, t-test and chi-square test, one-way ANOVA, and two-way ANOVA are used for both hypothesis and assessing significance testing in statistics.

Z-test

The z-test is a statistical test that compares the mean of a sample to the mean of a population or the means of two samples when the population standard deviation is known. It can determine if the difference between the means is statistically significant. For example, you can use a z-test to determine if the average height of students in your class differs from the average height of all students in your school, provided you know the standard deviation of the height of all students. To explain it simply, imagine you have two basketball teams, and you want to know if one team is taller than the other. You can measure the height of each player on both teams, calculate the average height for each team, and then use a z-test to determine if the difference between the averages is significant or just due to chance.

Tutorial 6.14: To illustrate the z-test test, based on above student height example, is as follows:

```
# import the ztest function from statsmodels package
from statsmodels.stats.weightstats import ztest
# create a list of heights (in cm) for each team
teamA = [180, 182, 185, 189, 191, 191, 192,
```

```
194, 199, 199, 205, 209, 209, 209, 210, 212, 212, 213, 214, 214]
teamB = [190, 191, 191, 191, 195, 195, 199, 199, 208, 209, 209, 214, 215, 216, 217, 217, 228, 229, 230, 233]
# perform a two sample z-test to compare the mean heights of the two teams
# the null hypothesis is that the mean heights are equal
# the alternative hypothesis is that the mean heights are different
# we use a two-tailed test with a significance level of 0.05
z_stat, p_value = ztest(teamA, teamB, value=0)
# print the test statistic and the p-value
print("Z-statistic:", z_stat)
print("P-value:", p_value)
# interpret the result
if p_value < 0.05:
    print("We reject the null hypothesis and conclude that the mean heights of the two teams are significantly different.")
else:
    print("We fail to reject the null hypothesis and conclude that the mean heights of the two teams are not significantly different.")
```

Output:

```
Z-statistic: -2.020774406815312
P-value: 0.0433030123323911 24
We reject the null hypothesis and conclude that the mean heights of the two teams are significantly different.
```

This means that, based on the sample data, there is enough evidence to suggest that Team B is, on average, taller than Team A, and that this difference is not due to chance.

T-test

A t-test is a statistical test that compares the mean of a sample to the mean of a population or the means of two samples. It can determine if the difference between the means is statistically significant or not, even when the population standard deviation is unknown and estimated from the sample. Here is a simple example: Suppose, you want to compare the delivery times of two different pizza places. You can order a pizza from each restaurant and record the time it takes for each pizza to arrive. Then, you can use a t-test to determine if the difference between the times is significant or if it could have occurred by chance. Another example is, you can use a t-test to determine whether *the average score of students who took a math test online differs from the average score of students who took the same test on paper*, provided that you are unaware of the standard deviation of the scores of all students who took the test.

Tutorial 6.15: To illustrate the t-test, based on above *student score* example, is as follows**:**

```
# import the ztest function from statsmodels package
from statsmodels.stats.weightstats import ztest
# create a list of delivery times (in minutes) for each piz
za place
placeA = [15, 18, 20, 22, 25, 28, 30, 32, 35, 40]
placeB = [12, 14, 16, 18, 20, 22, 24, 26, 28, 30]
# perform a two sample z-
test to compare the mean delivery times of the two pizza
 places
# the null hypothesis is that the mean delivery times are
 equal
# the alternative hypothesis is that the mean delivery ti
mes are
```

```
different
# we use a two-
tailed test with a significance level of 0.05
z_stat, p_value = ztest(placeA, placeB, value=0)
# print the test statistic and the p-value
print("Z-statistic:", z_stat)
print("P-value:", p_value)
# interpret the result
if p_value < 0.05:
    print("We reject the null hypothesis and conclude that the mean delivery times of the two pizza places are significantly different.")
else:
    print("We fail to reject the null hypothesis and conclude that the mean delivery times of the two pizza places are not significantly different.")
```

Output:

```
Z-statistic: 1.7407039045950503
P-value: 0.0817354935141978 6
We fail to reject the null hypothesis and conclude that th
e mean
delivery times of the two pizza places are not significantl
y different.
```

This means that based on the sample data, there is enough evidence to suggest that location B delivers faster than location A on average, and that this difference is not due to chance.

Chi-square test

The chi-square test is a statistical tool that compares observed and expected frequencies of categorical data under a null hypothesis. It can determine if there is a significant association between two categorical variables or

if the distribution of a categorical variable differs from the expected distribution. To determine if there is a relationship between the type of pet a person owns and their favorite color, or if the proportion of people who prefer chocolate ice cream is different from 50%, you can use a chi-square test.

Tutorial 6.16: Suppose, based on the above example of pets and favorite colors, you have data consisting of the observed frequencies of categories in *Table 6.1*, then implementation of the chi-square test on it, is as follows:

Pet	Red	Blue	Green	Yellow
Cat	12	18	10	15
Dog	8	14	12	11
Bird	5	9	15	6

***Table 6.1**: Pet a person owns, and their favorite color observed frequencies*

```
# import the chi2_contingency function
from scipy.stats import chi2_contingency
# create a contingency table as a list of lists
data = [[12, 18, 10, 15], [8, 14, 12, 11], [5, 9, 15, 6]]
# perform the chi-square test
stat, p, dof, expected = chi2_contingency(data)
# print the test statistic, the p-
value, and the expected frequencies
print("Test statistic:", stat)
print("P-value:", p)
print("Expected frequencies:")
print(expected)
# interpret the result
significance_level = 0.05
if p <= significance_level:
    print("We reject the null hypothesis and conclude that
```

```
    there is a significant association between the type of pet
    and the favorite color.")
else:
    print("We fail to reject the null hypothesis and conclu
    de that there is no significant association between the ty
    pe of pet and the favorite color.")
```

Output:

```
Test statistic: 6.740632143071166
P-value: 0.3455008329317587 6
Expected frequencies:
[[10.18518519 16.7037037  15.07407407 13.03703704]
 [ 8.33333333 13.66666667 12.33333333 10.66666667]
 [ 6.48148148 10.62962963  9.59259259  8.2962963 ]]
We fail to reject the null hypothesis and conclude that th
ere is no significant
association between the type of pet and the favorite colo
r.
```

Here, expected frequencies are the theoretical frequencies. We would expect to observe in each cell of the contingency table if the null hypothesis is true. They are calculated based on the row and column sums and the total number of observations. The chi-square test compares the observed frequencies (*Table 6.1*) with the expected frequencies (shown in the output) to see if there is a significant difference between them. Based on the sample data, there is insufficient evidence to suggest a correlation between a person's favorite color and the type of pet they own.

Another example is, to determine if a dice is fair, one can use the analogy of a dice game. You can roll the dice many times and count how many times each number comes up. You can use a chi-square test to determine if the observed counts are similar enough to the expected counts, which are equal for a fair dice, or if they differ too much to be attributed to chance. More about chi-square test is also in

Chapter 3, Measure of Association Section.

One-way ANOVA

A one-way ANOVA is a statistical test that compares the means of three or more groups that have been split on one independent variable. A one-way ANOVA can tell you if there is a significant difference among the group means or not. For example, you can use a one-way ANOVA to see if the average weight of dogs varies by breed, if you have data on the weight of dogs from three or more breeds. Another example is, you can use an analogy of a baking contest to know if the type of flour you use affects the taste of your cake. You can bake three cakes using different types of flour and ask some judges to rate the taste of each cake. Then you can use a one-way ANOVA to see if the average rating of the cakes is different depending on the type of flour, or if they are all similar.

Tutorial 6.17: To illustrate the one-way ANOVA test, based on above baking contest example, is as follows.

```
import numpy as np
import scipy.stats as stats
# Define the ratings of the cakes by the judges
cake1 = [8.4, 7.6, 9.2, 8.9, 7.8] # Cake made with flour type 1
cake2 = [6.5, 5.7, 7.3, 6.8, 6.4] # Cake made with flour type 2
cake3 = [7.1, 6.9, 8.2, 7.4, 7.0] # Cake made with flour type 3
# Perform one-way ANOVA
f_stat, p_value = stats.f_oneway(cake1, cake2, cake3)
# Print the results
print("F-statistic:", f_stat)
print("P-value:", p_value)
```

Output:

1. F-statistic: 11.716117216117217
2. P-value: 0.0015090242950033377

The p-value is very small, which means that we can reject the null hypothesis that the means of the ratings are equal. This suggests that the type of flour affects the taste of the cake.

Two-way ANOVA

A two-way ANOVA is a statistical test that compares the means of three or more groups split on two independent variables. It can determine if there is a significant difference among the group means, if there is a significant interaction between the two independent variables, or both. For example, if you have data on the blood pressure of patients from different genders and age groups, you can use a two-way ANOVA to determine if the average blood pressure of patients varies by gender and age group. Another example is, analogy of a science fair project. Imagine, you want to find out if the type of music you listen to and the time of day you study affect your memory. Volunteers can be asked to memorize a list of words while listening to different types of music (such as classical, rock, or pop) at various times of the day (such as morning, afternoon, or evening). Their recall of the words can then be tested, and their memory score measured. A two-way ANOVA can be used to determine if the average memory score of the volunteers differs depending on the type of music and time of day, or if there is an interaction between these two factors. For instance, it may show, listening to classical music may enhance memory more effectively in the morning than in the evening, while rock music may have the opposite effect.

Tutorial 6.18: The implementation of two-way ANOVA test, based on above baking contest example, is as follows**:**

```
import pandas as pd
import statsmodels.api as sm
from statsmodels.formula.api import ols
from statsmodels.stats.anova import anova_lm
# Define the data
data = {"music": ["classical", "classical", "classical", "classical", "classical",
                  "rock", "rock", "rock", "rock", "rock",
                  "pop", "pop", "pop", "pop", "pop"],
        "time": ["morning", "morning", "afternoon", "afternoon", "evening",
                 "morning", "morning", "afternoon", "afternoon", "evening",
                 "morning", "morning", "afternoon", "afternoon", "evening"],
        "score": [12, 14, 11, 10, 9,
                  8, 7, 9, 8, 6,
                  10, 11, 12, 13, 14]}
# Create a pandas DataFrame
df = pd.DataFrame(data)
# Perform two-way ANOVA
model = ols("score ~ C(music) + C(time) + C(music):C(time)", data=df).fit()
aov_table = anova_lm(model, typ=2)
# Print the results
print(aov_table)
```

Output:

```
                    sum_sq   df         F    PR(>F)
C(music)         54.933333  2.0  36.622222  0.000434
C(time)           1.433333  2.0   0.955556  0.436256
C(music):C(time)  24.066667  4.0   8.022222  0.013788
Residual          4.500000  6.0        NaN       NaN
```

Since the p-value for music is less than 0.05, the music has a significant effect on memory score, while time has no significant effect. And since the p-value for the interaction effect (0.013788) is less than 0.05, this tells us that there is a significant interaction effect between music and time.

Hypothesis and significance testing in diabetes dataset

Let us use the diabetes dataset, containing information on 768 patients. Out of it, let us take **body mass index** (**BMI**) and outcome (whether they have diabetes or not) where 0 means no diabetes and 1 means diabetes.

Now, to perform testing, we will define a research question in the form of a hypothesis, as follows:

- **Null hypothesis:** The mean BMI of diabetic patients is equal to the mean BMI of non-diabetic patients.
- **Alternative hypothesis:** The mean BMI of diabetics is not equal to the mean BMI of non-diabetics.

Tutorial 6.19: The implementation of hypothesis testing and significance on diabetes dataset to test is as follows:

```
import pandas as pd
from scipy import stats
# Load the diabetes data from a csv file
data = pd.read_csv("/workspaces/ImplementingStatisticsWithPython/data/chapter1/diabetes.csv")
# Null hypothesis: There is a significant difference in the mean BMI of diabetic and non-diabetic patients
# Separate the BMI values for diabetic and non-diabetic patients
bmi_diabetic = data[data["Outcome"] == 1]["BMI"]
bmi_non_diabetic = data[data["Outcome"] == 0]["BMI"]
# Perform a two-sample t-test to compare the means of the two groups
```

```
t, p = stats.ttest_ind(bmi_diabetic, bmi_non_diabetic)
# Print the test statistic and the p-value
print("Test statistic:", t)
print("P-value:", p)
# Set a significance level
alpha = 0.05
# Compare the p-
value with the significance level and make a decision
if p <= alpha:
    print("We reject the null hypothesis and conclude that
 there is a significant difference in the mean BMI of diab
etic and non-diabetic patients.")
else:
    print("We fail to reject the null hypothesis and conclu
de that there is not enough evidence to support a signifi
cant difference in the mean BMI of diabetic and non-
diabetic patients.")
```

Output:

```
Test statistic: 8.47183994786525
P-value: 1.229807487311602e-16
We reject the null hypothesis and conclude that there is
a significant
difference in the mean BMI of diabetic and non-
diabetic patients.
```

The output shows the mean BMI of diabetics is not equal to the mean BMI of non-diabetics, which means the BMI of diabetic and non-diabetic person is different.

Tutorial 6.20: To measure if there is an association between the number of pregnancies and the outcome, we define **null hypothesis**: *There is no association between the number of pregnancies and the outcome (diabetic and non-diabetic patients).* **Alternative hypothesis:** *There is association between the number of pregnancies and the*

outcome (diabetic and non-diabetic patients). Then the implementation of hypothesis testing and the significance on diabetes dataset, is as follows:

```
import pandas as pd
from scipy import stats
# Load the diabetes data from a csv file
data = pd.read_csv("/workspaces/ImplementingStatisticsWithPython/data/chapter1/diabetes.csv")
# Separate the number of pregnancies and the outcome for each patient
pregnancies = data["Pregnancies"]
outcome = data["Outcome"]
# Perform a chi-square test to test the independence of the two variables
chi2, p, dof, expected = stats.chi2_contingency(pd.crosstab(pregnancies, outcome))
# Print the test statistic and the p-value
print("Test statistic:", chi2)
print("P-value:", p)
# Set a significance level
alpha = 0.05
# Compare the p-value with the significance level and make a decision
if p <= alpha:
    print("We reject the null hypothesis and conclude that there is a significant association between the number of pregnancies and the outcome.")
else:
    print("We fail to reject the null hypothesis and conclude that there is not enough evidence to support a significant association between the number of pregnancies and the outcome.")
```

Output:

1. Test statistic: 64.59480868723006
2. P-value: 8.648349123362548e-08
3. We reject the null hypothesis and conclude that there is a significant association
   between the number of pregnancies and the outcome.

Sampling techniques and sampling distributions

Sampling techniques involve selecting a subset of individuals or items from a larger population. Sampling distributions display how a sample statistic, such as the mean, proportion, or standard deviation, varies across many random samples from the same population. These techniques and distributions are used in statistics to make inferences or predictions about the entire population based on the sample data. To determine the average height of all students in your school, measuring each student's height would be impractical and time-consuming. Instead, you can use a sampling technique, such as simple random sampling, to select a smaller group of students, for example 100, and measure their heights. This smaller group is called a **sample**, and the average height of this sample is called a **sample mean**.

Imagine repeating this process multiple times, selecting a different random sample of 100 students each time, and calculating their average height. Each sample is different, resulting in different sample means. Plotting all these sample means on a graph creates a sampling distribution of the sample mean. This graph will show how the sample mean varies across different samples and the most likely value of the sample mean.

The sampling distribution of the sample mean has several interesting properties. One of these is that its mean is equal to the population mean. This implies that the average of all the sample means is the same as the average of all the students in the school. Additionally, the shape of the

sampling distribution of the sample mean approaches a bell curve (also known as a normal distribution) as the sample size increases. The central limit theorem enables us to use the normal distribution to predict the population mean based on the sample mean.

Tutorial 6.21: A simple illustration of the sampling technique using 15 random numbers, is as follows:

```
import random
# Sampling technique
data = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]
sample_size = 5
sample = random.sample(data, sample_size)
print(f"The sample of size {sample_size} is: {sample}")
```

Output:

```
The sample of size 5 is: [8, 11, 9, 14, 4]
```

Tutorial 6.22: A simple illustration of the sampling distribution using 1000 samples of size 5 generated from a list of 15 integers. We then calculate the mean of each sample and store it in a list, as follows:

```
import random
# Sampling distribution
sample_size = 5
num_samples = 1000
data = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15]
sample_means = []
for i in range(num_samples):
    sample = random.sample(data, sample_size)
    sample_mean = sum(sample) / sample_size
    sample_means.append(sample_mean)
print(f"The mean of the sample means is: {sum(sample_means) / num_samples}")
```

Output:

```
The mean of the sample means is: 8.006000000000002
```

Further to understand it in simple words, let us take another example of *rolling dice*. To determine the average number of dots when rolling a die, we must first define the population as the set of all possible outcomes: To determine the average number of dots when rolling a die, we must first define the population as the set of all possible outcomes: To determine the average number of dots when rolling a die, we must first define the population as the set of all possible outcomes: 1, 2, 3, 4, 5, and 6. To determine the average number of dots when rolling a die, we must first define the population as the set of all possible outcomes. By using this sample, we can estimate the population mean. In case of rolling a die population mean is 3.5, however, since it is impossible to roll a 3.5, we need to use a sample to estimate it. One method is to roll the die once and record the number of dots. This is a sample of size 1. The sample mean is equal to the number of dots. If you repeat this process multiple times, you will obtain different sample means each time, ranging from 1 to 6.

Plotting these sample means on a graph will result in a sampling distribution of the sample mean that appears as a flat line, with equal chances of obtaining any number from 1 to 6. However, this sampling distribution is not very informative as it does not provide much insight into the population mean. One way to obtain a sample of size 2 is by rolling a die twice and adding up the dots. The sample mean is then calculated by dividing the sum of the dots by 2. If this process is repeated multiple times, different sample means will be obtained, each with a probability of occurrence. The probabilities range from 1 to 6, depending on the sample mean. For instance, the probability of obtaining a sample mean of 2 is 1/36, as it requires rolling two ones, which has a probability of 1/6 multiplied by 1/6. The probability of obtaining a sample mean of 3 is 2/36. This is because you can roll a one and a two, or a two and a one, which has a probability of 2/6 times 1/6.

If you plot these sample means on a graph, you will get a sampling distribution of the sample mean that looks like a triangle. The distribution has higher chances of obtaining numbers closer to 3.5. This sampling distribution is more useful because it indicates that the population mean is more likely to be around 3.5 than around 1 or 6. To increase the sample size, roll a die three or more times and calculate the sample mean each time. As the sample size increases, the sampling distribution of the sample mean becomes more bell-shaped, with a narrower and taller curve, indicating greater accuracy and consistency. The central limit theorem is demonstrated here, allowing you to predict the population mean using the normal distribution based on the sample mean.

For instance, if you roll a die 30 times and obtain a sample mean of 3.8, you can use the normal distribution to determine the likelihood that the population mean falls within a specific range of 3.5 to 4.1. This is a confidence interval. It provides an idea of how certain you are that your sample mean is close to the population mean. The confidence interval becomes narrower with a larger sample size, increasing your confidence.

Tutorial 6.23: To explore sampling distributions and confidence intervals through dice rolls, is as follows:

```
1. import numpy as np
2. import matplotlib.pyplot as plt
3. # Roll Dice
4. def roll_die(num_rolls):
5.     return np.random.randint(1, 7, num_rolls)
6. # Function to generate sample means for rolling dice
7. def dice_sample_means(num_rolls, num_samples):
8.     means = []
9.     for _ in range(num_samples):
10.         sample = roll_die(num_rolls)
```

```
        means.append(np.mean(sample))
    return means
# Generate sampling distribution for rolling a die
num_rolls = 30
num_samples = 1000
dice_means = dice_sample_means(num_rolls, num_samples)
# Convert dice_means to a NumPy array
dice_means = np.array(dice_means)
# Plotting the sampling distribution of the sample mean for dice rolls
plt.figure(figsize=(10, 6))
plt.hist(dice_means, bins=30, density=True, alpha=0.6, color='b')
plt.axvline(3.5, color='r', linestyle='--')
plt.title('Sampling Distribution of the Sample Mean (Dice Rolls)')
plt.xlabel('Sample Mean')
plt.ylabel('Frequency')
plt.show()
# Confidence Interval Example
sample_mean = np.mean(dice_means)
sample_std = np.std(dice_means)
# Calculate 95% confidence interval
conf_interval = (sample_mean - 1.96 * (sample_std / np.sqrt(num_rolls)),
                 sample_mean + 1.96 * (sample_std / np.sqrt(num_rolls)))
print(f"Sample Mean: {sample_mean}")
print(f"95% Confidence Interval: {conf_interval}")
```

Output:

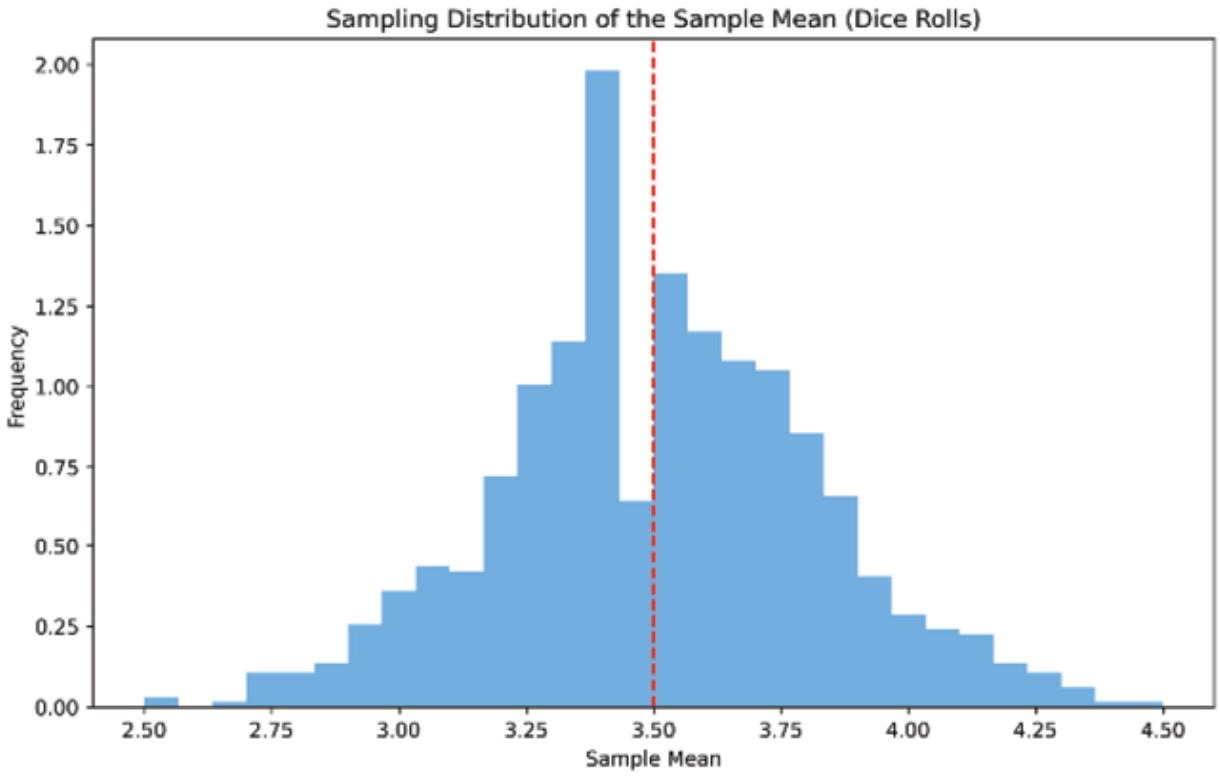

***Figure 6.1**: Sampling distribution of the sample mean*

Conclusion

In this chapter, we learned about the concept and process of hypothesis testing, which is a statistical method for testing whether or not a statement about a population parameter is true. Hypothesis testing is important because it allows us to draw conclusions from data and test the validity of our claims.

We also learned about significance tests, which are used to evaluate the strength of evidence against the null hypothesis based on the p-value and significance level. Significance testing uses the p-value and significance level to determine whether the observed effect is statistically significant, meaning that it is unlikely to occur by chance. We explored different types of statistical tests, such as z-test, t-test, chi-squared test, one-way ANOVA, and two-way

ANOVA, and how to choose the appropriate test based on the research question, data type, and sample size. We also discussed the importance of sampling techniques and sampling distributions, which are essential for conducting valid and reliable hypothesis tests. To illustrate the application of hypothesis testing, we conducted two examples using a diabetes dataset. The first example tested the null hypothesis that the mean BMI of diabetic patients is equal to the mean BMI of non-diabetic patients using a two-sample t-test. The second example tests the null hypothesis that there is no association between the number of pregnancies and the outcome (diabetic versus non-diabetic) using a chi-squared test.

Chapter 7, Statistical Machine Learning discusses the concept of machine learning and how to apply it to make artificial intelligent models and evaluate them.

Join our book’s Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

https://discord.bpbonline.com